



Aula 04 - Apresentação da plataforma ESP32

Módulo de Internet das Coisas

- Prof.

MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO

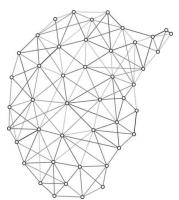


Softex



MCTI
FUTURO

IA



Objetivos da Aula

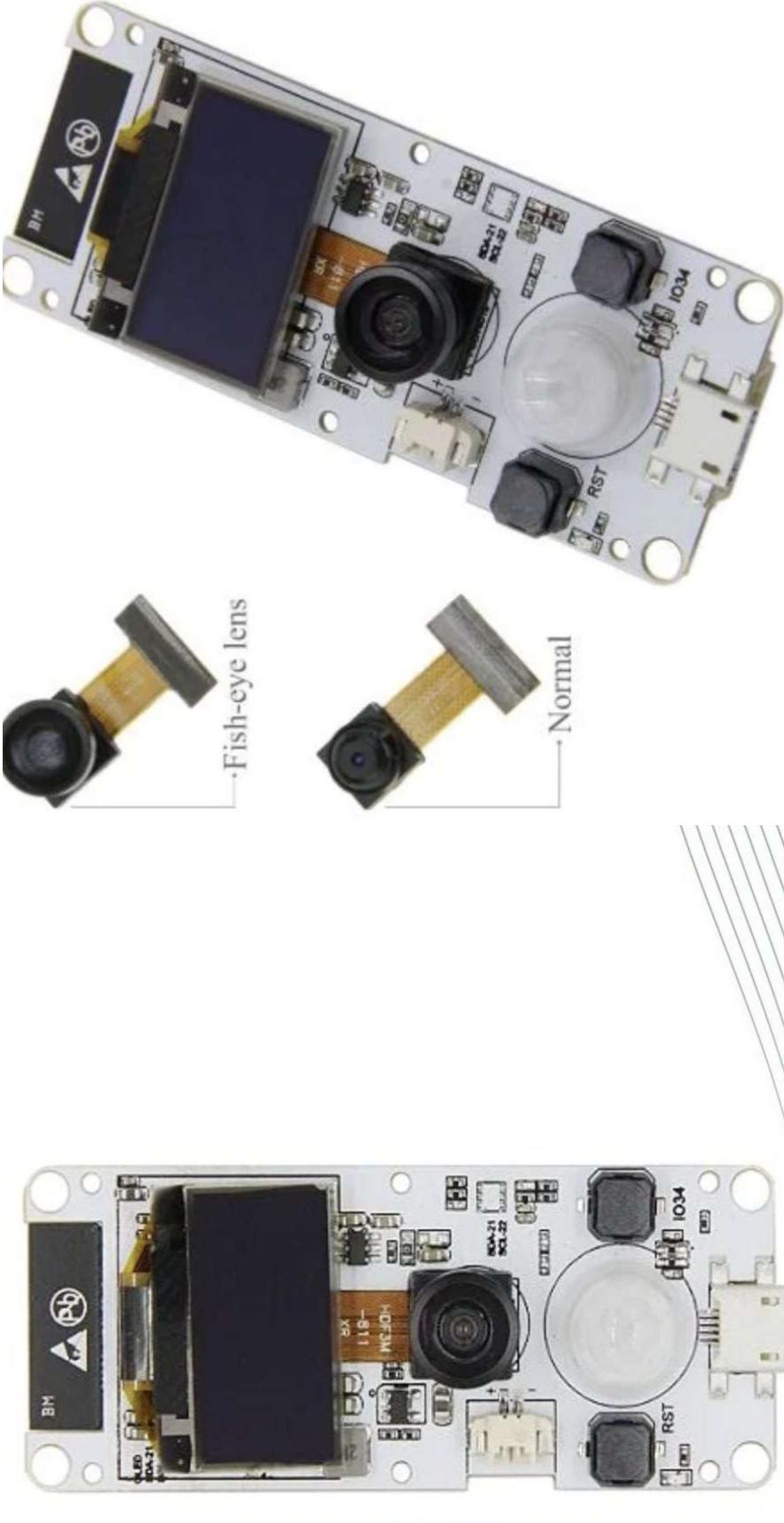


- Introdução ao ESP32
- Revisão de placas de desenvolvimento baseadas no ESP32
- Configuração de um ambiente de desenvolvimento
- Construção de programas com Espressif SDK
- Desenvolvimento de programas Sketch no ESP32

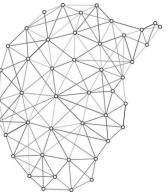
IA

Requisitos técnicos

- Um computador com um SO instalado como Windows, Linux ou macOS
- Uma placa de desenvolvimento ESP32 cam TTGO

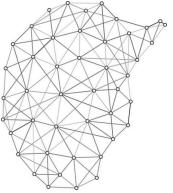


IA



Introdução ao ESP32-WROVER-B Wireless

- Chip: ESP32 Dual Core Tensilica LX6 processor
- Memória: 8MB PSRAM
- Armazenamento: 4MB SPI Flash
- Conectividade: 2.4 GHz 802.11 b/g/n WiFi 5, Bluetooth 4.2 LE
- Câmera: 2MP OV2640 Fish-Eye Lens
- Display: 0.96" 128x64 OLED
- Controlador do Display: SSD1306 I_C Display Controller
- Porta Micro USB para alimentação e programação (CP2104)
- Sensor de Movimento: AS312 PIR
- Conectores de Expansão: I_C (SDA, SDA), GND, CON +3,3V e 5V
- Botões: Reset e Programável (IO34)
- Alimentação: 5V via micro USB ou bateria LiPo via 2-pin e o chip de controlador IP5306 integrado
- Tamanho: 68mm Largura x 28mm Profundidade x 20mm Altura



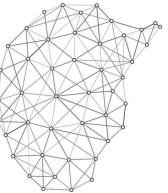
IIA

Revisão de placas de desenvolvimento baseada na ESP32

- O ESP32 possui duas formas: chip e módulo
- Neste curso, não vamos aprender como criar uma placa baseada no ESP32
- Ao invés disso, vamos usar uma placa de desenvolvimento disponível no mercado

- Existem duas categorias de placa ESP32
 - manufaturados pela Espressif
 - manufaturadas pelos seus parceiros e personal makers

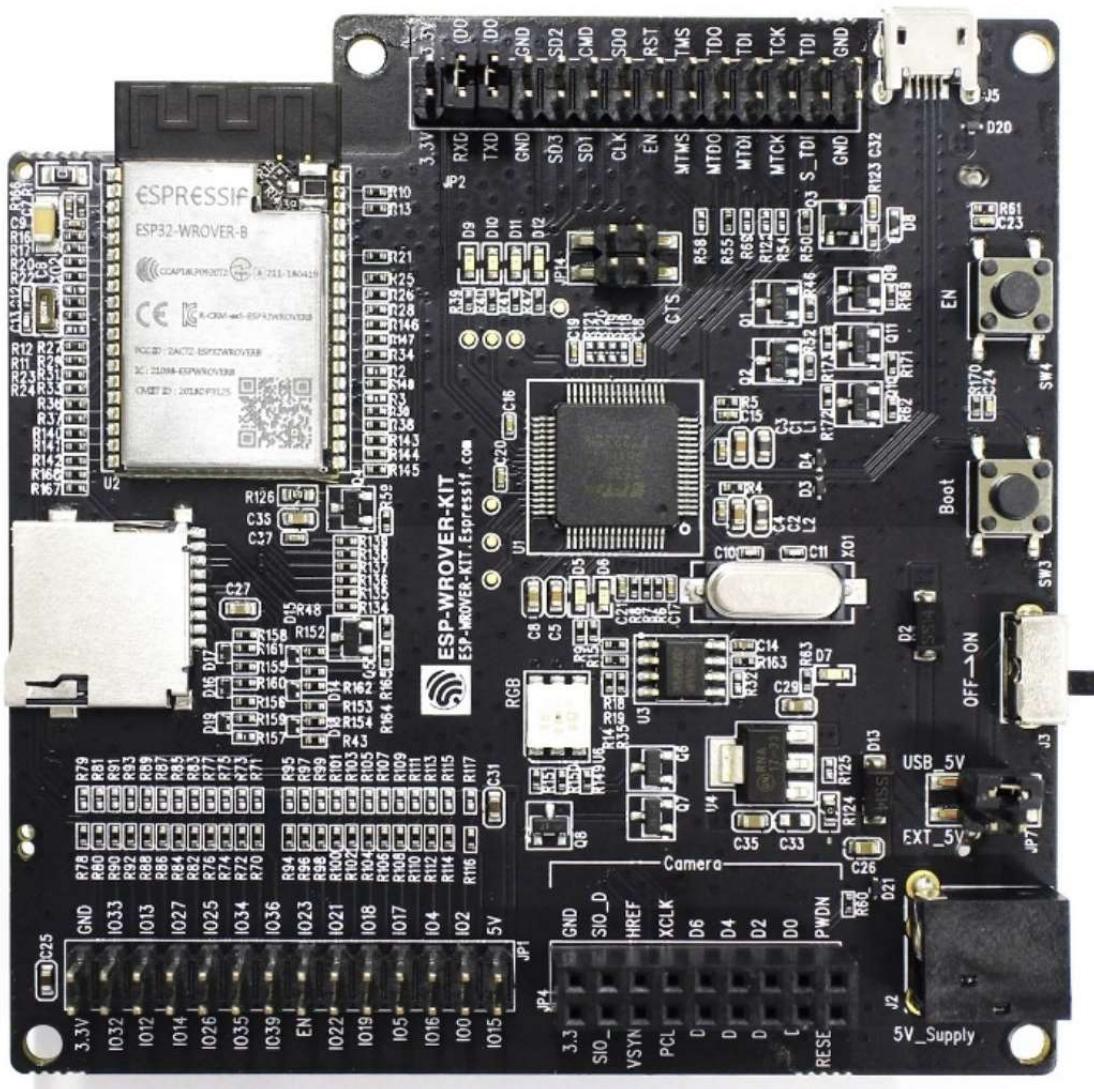
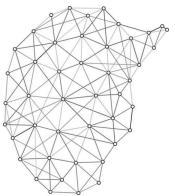
IA



O kit de desenvolvimento oficial do ESP32

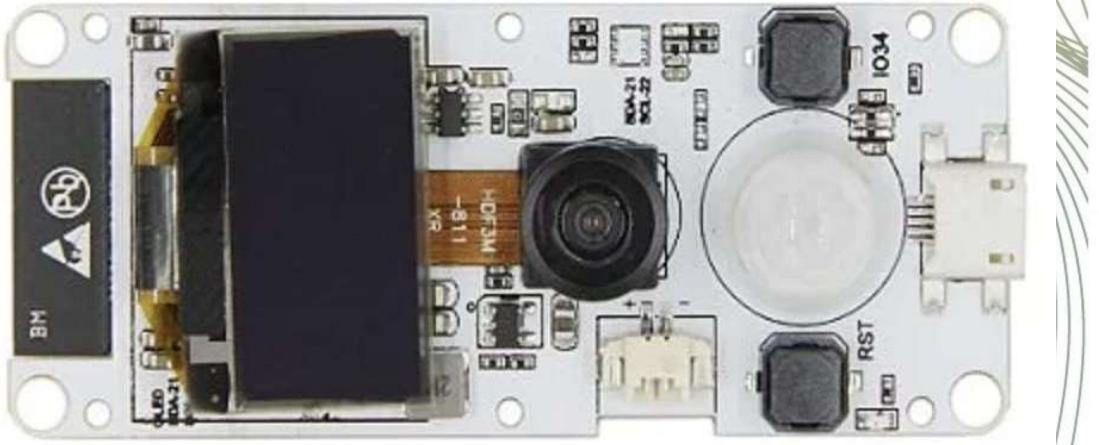
- Kits que podemos usar diretamente sem precisar fazer uma placa PCB a soldagem de um chip ESP32.
- **ESP32-PICO-KIT**
 - desenvolvimento básico, tamanho pequeno
 - cabe numa breadboard PCB, então dá para você fazer a fiação/ligaçāo
 - consiste em chips ESP32 como o serial USB CP2102/CP2102N
 - conexāo USB com o computador
- **ESP-WROVER-KIT**
 - desenvolvimento completo
 - consiste em vários sensores e módulos
 - Principais características: interface JTAG na FT2232HL, conector camera, conexāo I/O, RGB LED, slot para Micro SD card, LCD

ESP-WROVER-KIT



Camera

y9	—	39
y8	—	36
y7	—	23
y6	—	18
y5	—	15
y4	—	4
y3	—	14
y2	—	5
VSNC	—	27
HREF	—	25
PCLK	—	19
PWD	—	26
XCLK	—	32
SIOD	—	13
SIOC	—	12
RST	—	NC



MCTI
FUTURO

Instituto Iracema
PESQUISA E INovação



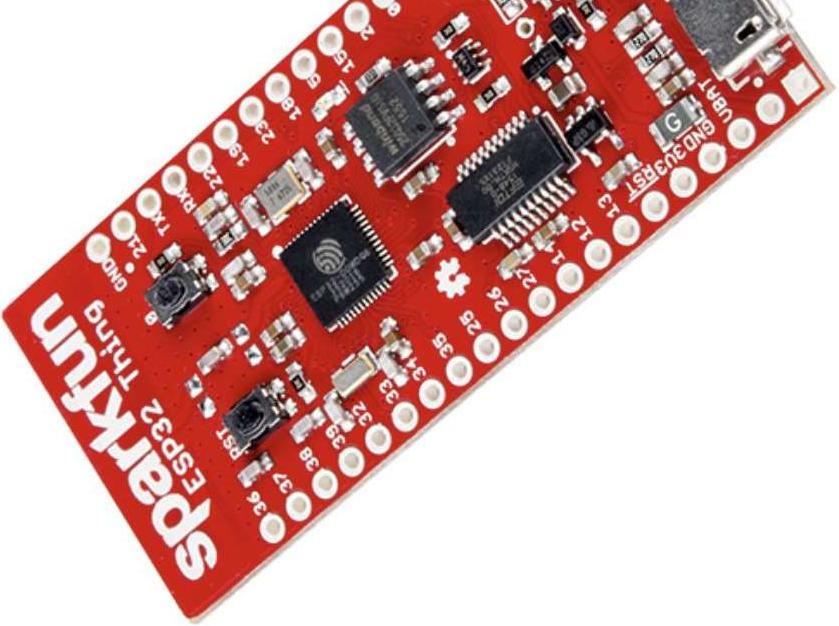
MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO

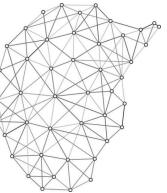
GOVERNO
BRASIL
UNIÃO E RE

IA

Placas baseadas na ESP32 de Terceiros

- Chips e módulos ESP32 podem ser comprados na Espressif e seus distribuidores para a criação de suas de placas de desenvolvimento
- SparkFun ESP32 Thing
 - vendida pela SparkFun
 - usa chip ESP32
 - provê TTL USB para conectar com o chip ESP32
 - tem conector LiPo para rodar com bateria

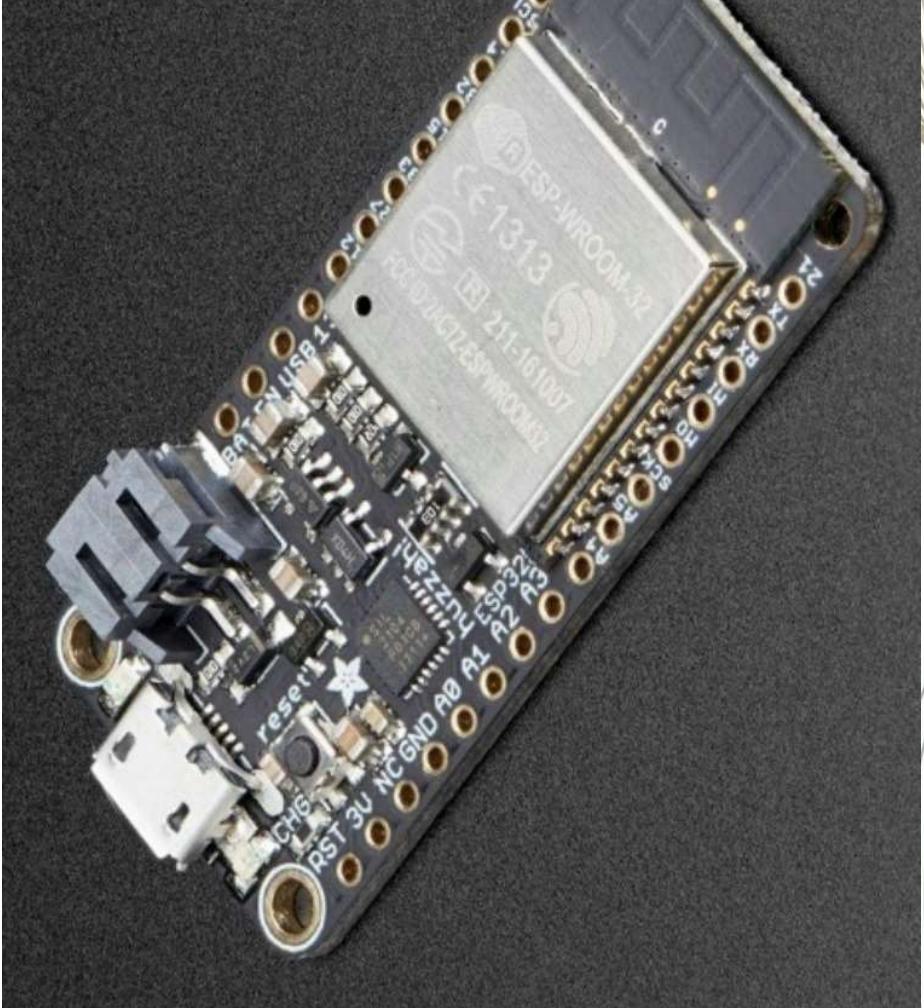


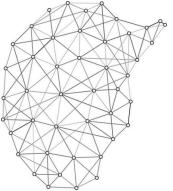


IA

Placas baseadas na ESP32 de Terceiros

- Adafruit HUZZAH32 - ESP32 Feather Board
 - vendida pela Adafruit pela Internet
 - usa módulo ESP32
 - possui TTL USB e conector LiPo
- Outras placas podem ser encontradas na Alibaba e Aliexpress





IIA

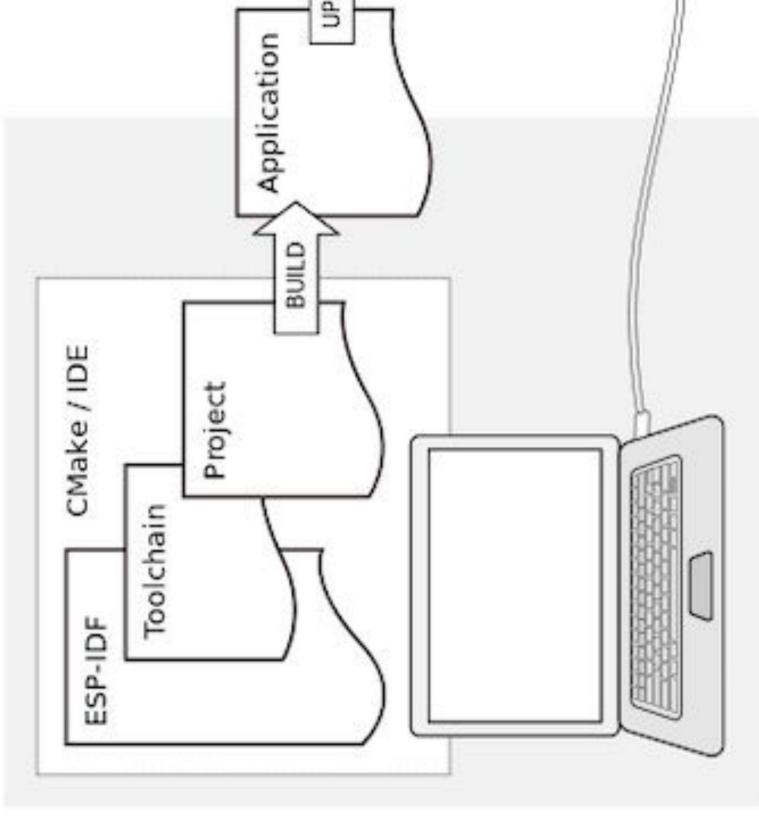
Configuração de um ambiente de desenvolvimento

- Toolchain para compilar o código para ESP32
 - [como configurar a ESP32 toolchain no Windows, Linux, macOS](#)

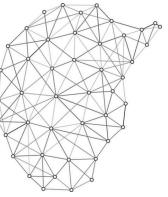
- ESP-IDF habilita o desenvolvimento do programa ESP32
 - contém API (bibliotecas e código fonte)
 - scripts para operar a toolchain

- Build tools
 - CMAKE e Ninja para construir uma aplicação completa para ESP32

- [Visual Studio Code](#)
 - IDE tool para escrever programas em C

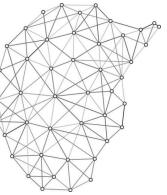


IA



Demo 1 - Construir seu primeiro programa ESP32

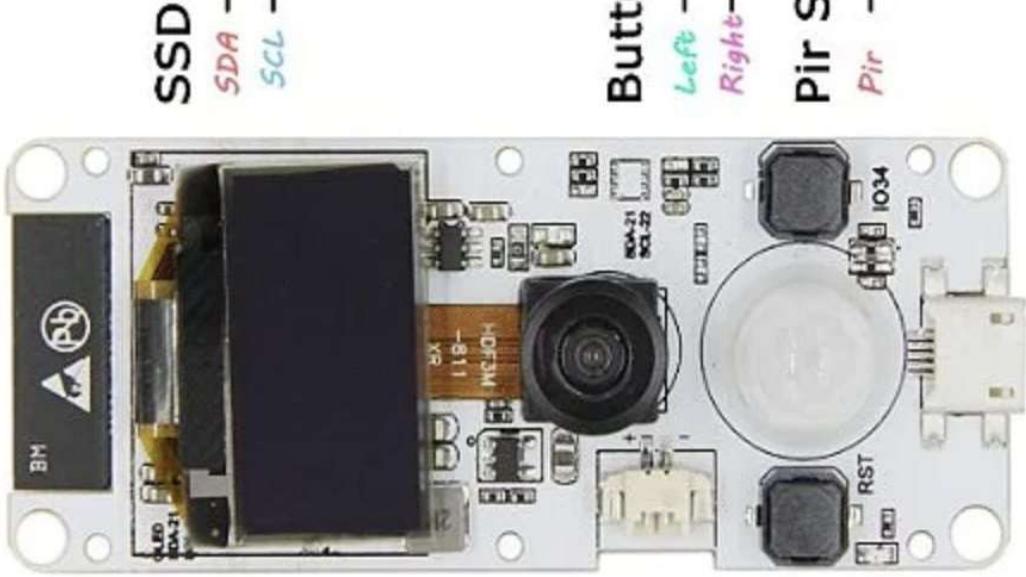
- PIR Sensor Read: escrever na serial a leitura de um sensor de movimento
 - Criar um projeto
 - Escrever um programa
 - Configurar o projeto
 - Compilar e gravar



IA

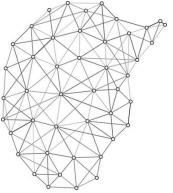
Deteção de Movimento com o PIR AS312 Integrado

- O PIR sensor está conectado ao pin 33 e lê:
 - 0, quando o movimento é baixo
 - 1, quando o movimento é alto



Camera

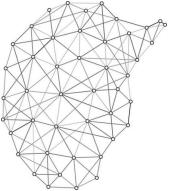
Y9	—	39
Y8	—	36
Y7	—	23
Y6	—	18
Y5	—	15
Y4	—	4
Y3	—	14
Y2	—	5
VSNC	—	27
HREF	—	25
PCLK	—	19
PWD	—	26
XCLK	—	32
SIOD	—	13
SIOC	—	12
RST	—	NC



IA

Demo 1 - PIR Sensor Read- Criar um projeto

- Não existe um projeto template para o programa ESP32 com SDK
 - O programa pode ter a estrutura da figura
 - Cada projeto tem os seguintes arquivos:
 - **Makefile** na raiz do projeto
 - pasta main
 - arquivo do programa (*.c)
 - arquivo **component.mk** dentro da pasta main
 - Vamos criar na sequência:
 - a pasta chamada **pirsensoread**
 - arquivo **Makefile**
 - pasta **main**
 - Dentro da main, os arquivos **pirsensoread.c** e **component.mk**



IA

Demo 1 - PIR Sensor Read- Escrever o programa (1/4)

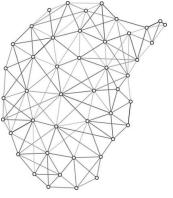
Vamos escrever scripts e códigos nos arquivos **Makefile**, **component.mk** e **pirsensoread.c**

1. No arquivo **Makefile**, deve-se declarar o nome do projeto com o mesmo nome da pasta

```
PROJECT_NAME := pirsensoread
include $(IDF_PATH)/make/project.mk
```

2. **component.mk** é necessário para a compilação e deve ser criado esse exato nome. O conteúdo do **component.mk** é vazio.

```
# "main" pseudo-component makefile.
#
# (Uses default behavior of compiling all source files in
# directory, adding 'include' to include path.)
```



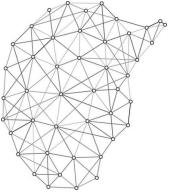
IA

Demo 1 - PIR Sensor Read- Escrever o programa (2/4)

1. Na main.c, nós importamos as bibliotecas e declaramos o Pir Sensor pino 33 do ESP32

```
#include "freertos/Freertos.h"  
#include "freertos/task.h"  
#include "esp_system.h"  
#include "driver/gpio.h"  
#include "esp_log.h"
```

```
#define PIR_GPIO 33
```



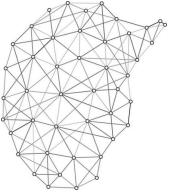
IA

Demo 1 - PIR Sensor Read- Escrever o programa (3/4)

Ainda no arquivo `main.c` (continuação)

A entrada principal do programa é `app main()`. Nela, vamos inicializar definir o tipo de periférico que vamos utilizar, nesse caso nosso sensor PIR input. Iremos adicionar um loop semelhante a função `loop()` na arduino IDE mas nesse caso com um `while(1)`, que logo após as initializações anteriores continuará executando o que está em seu conteúdo “infinitamente” ou até alguma interrupção.

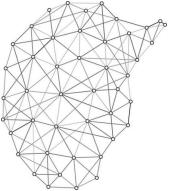
No conteúdo do nosso loop, será adicionado uma condição que se `gpio_get_level(PIN_GPIO)` seja verdade, ou seja, se o nível correspondente ao Sensor PIR for 1, será exibido no console serial a seguinte mensagem: “Detectado movimento”. Caso não, será exibido a mensagem “Nenhum movimento”. Após isso, damos um delay de 1000 milissegundos.



IA

Demo 1 - Blinking - Escrever o programa (3/4 cont.)

```
void app_main(void) {  
    esp_rom_gpio_pad_select_gpio(PIR_GPIO);  
    gpio_set_direction(PIR_GPIO, GPIO_MODE_INPUT);  
  
    while(1){  
  
        if(gpio_get_level(PIR_GPIO))  
        {  
            printf("\n --Detectado movimento-- \n");  
        }  
        else  
        {  
            printf("\n --Nenhum movimento-- \n");  
        }  
        vTaskDelay(1000/portTICK_PERIOD_MS);  
    }  
}
```

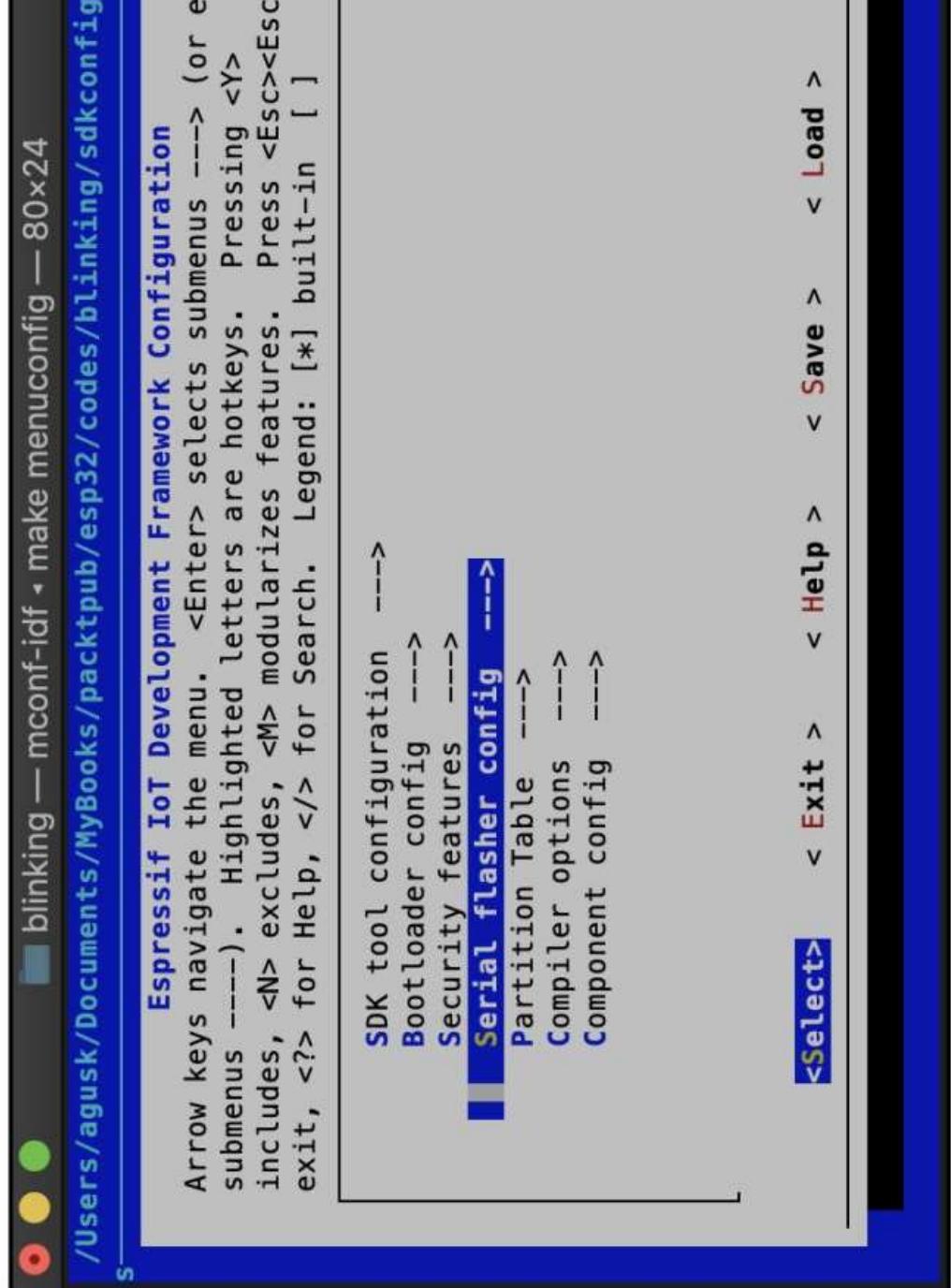


IA

Demo 1 - PIR Sensor Read- Configurar o projeto (1/3)

Configurar o projeto
usando **menuconfig**,
ferramenta da ESP32
toolchain

1. Abra o terminal e navegue até o diretório do seu projeto. Depois, digite o seguinte comando
make menuconfig
2. O diálogo ao lado será mostrado. Selecione o menu **Serial flasher config**.



IA

Demo 1 - PIR Sensor Read- Configurar o projeto (2/3)

1. Aqui, preencha com a porta serial da sua placa ESP32.

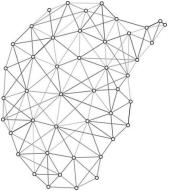
```
blinking — mconf-idf - make menuconfig — 80x24
/Users/agusk/Documents/MyBooks/packtpub/esp32/codes/blinking/sdkconfig - Express
s-> Serial flasher config
```

Default serial port

Please enter a string value. Use the <TAB> key to move from the input field to the buttons below it.

/dev/tty.usbserial-00002014B

< Ok > < Help >



IA

Demo 1 - PIR Sensor Read- Configurar o projeto (3/3)

1. Clique no botão  para iniciar o projeto.
- Save. O programa **menuconfig** vai salvar a configuração do seu projeto. Esta ferramenta gera o arquivo **sdkconfig** no diretório do seu projeto.

blinking — -bash — 80x24

```
/esp/esp-idf/tools/kconfig/lxdialog/yesno.c -o lxdialog/yesno.o  
cc -c -D_DARWIN_C_SOURCE -I/opt/local/include -DCURSES_LOC=<ncurses  
SES_WIDECHAR=1 -DKBUILD_NO_NLS -Wno-format-security -DLocale -MMD /  
esp/esp-idf/tools/kconfig/lxdialog/menubox.c -o lxdialog/menubox.o  
cc -o mconf-idf mconf.o zconf.tab.o lxdialog/checklist.o lxdialog/util  
g/inputbox.o lxdialog/textbox.o lxdialog/yesno.o lxdialog/menubox.o -  
/opt/local/lib -lncurses  
cc -c -D_DARWIN_C_SOURCE -I/opt/local/include -DCURSES_LOC=<ncurses  
SES_WIDECHAR=1 -DKBUILD_NO_NLS -Wno-format-security -DLocale -MMD /  
esp/esp-idf/tools/kconfig/conf.c -o conf.o  
cc -o conf-idf conf.o zconf.tab.o -lncurses -L/opt/local/lib -lncurses  
DEFCONFIG  
# configuration written to /Users/agusk/Documents/MyBooks/packtpub/es  
linking/sdkconfig  
#  
MENUCONFIG  
  
*** End of the configuration.  
*** Execute 'make' to start the build or try 'make help'.  
GENCONFIG  
agusk$
```

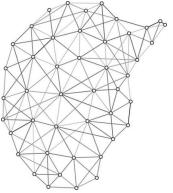


Instituto Iracema
PESQUISA E INovação



MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO





IA

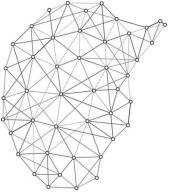
Demo 1 - PIR Sensor Read- Compilar e gravar

- Para gravar o programa na placa ESP32, use o comando **make flash** no terminal dentro do diretório dentro do diretório dentro do projeto.
- Se tudo der certo, você verá a detecção de movimento na serial

```
● ● ●
blinking — -bash — 80x24

None
MAC: 30:ae:a4:ef:4b:e8
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Flash params set to 0x0220
Compressed 22992 bytes to 13646...
Wrote 22992 bytes (13646 compressed) at 0x000001000 in 1.2 seconds (effe
.3 kbit/s)... .
Hash of data verified.
Compressed 155488 bytes to 74039...
Wrote 155488 bytes (74039 compressed) at 0x00010000 in 6.6 seconds (e
8.7 kbit/s)... .
Hash of data verified.
Compressed 3072 bytes to 103...
Wrote 3072 bytes (103 compressed) at 0x00008000 in 0.0 seconds (effe
kbit/s)... .
Hash of data verified.

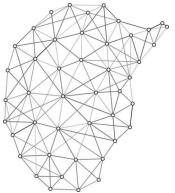
Leaving...
Hard resetting via RTS pin...
agusk$
```



IA

Demo 2 - Fazer um programa Arduino sketch para ES

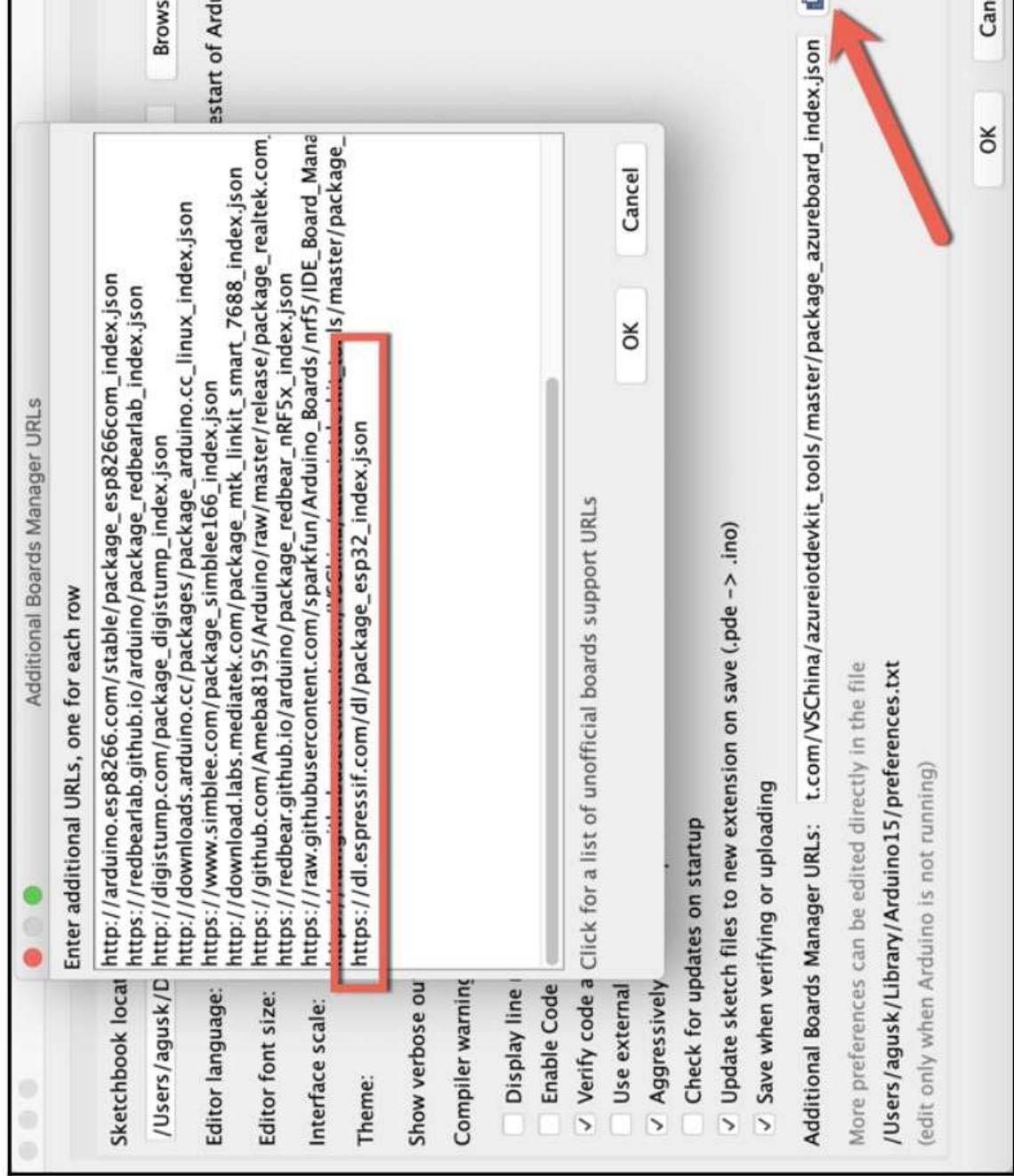
- Arduino é a maior comunidade para hardware open source
- Existem várias placas Arduino para suas necessidades
- Arduino também prover software para desenvolver o programa Arduino Sketch -> [Download](#)
- Placas ESP32 agora suportam desenvolvimento Arduino com o Expresso SDK. Você deve configurar o software Arduino Sketch para permitir que você trabalhe com placas ESP32-> [HowTo.](#)



IA

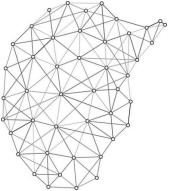
Demo 2 - Fazer um programa Arduino sketch para ES

1. Recomenda-se instalar as placas **ESP32** no software **Arduino** via **Board Manager**. Abra o diálogo **Preferences** do **Arduino** e ponha **esta URL** no board manager de URL.
2. Depois, clique em **OK**.



Instituto Iracema
PESQUISA E INOVAÇÃO

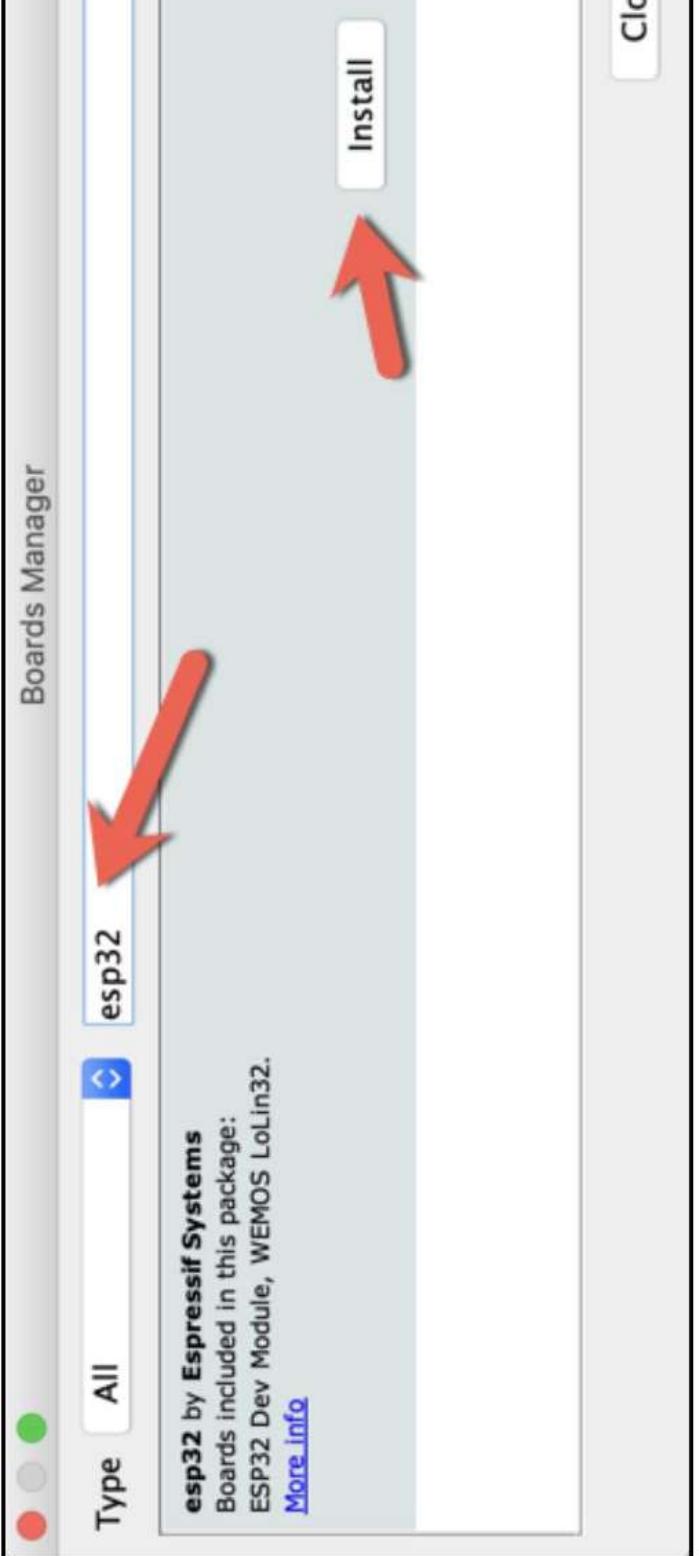




IA

Demo 2 - Fazer um programa Arduino sketch para ES

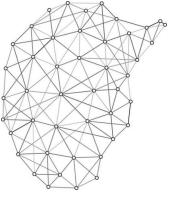
1. Para instalar as placas ESP32, abra o Boards Manager no menu Tool. Dignite **esp32** no form para que você veja o pacote **esp32** como most abaixo.
2. Depois clique em **Install**, para o Arduino baixar as bibliotecas necessaria a ESP32.



MCTI
Instituto Iracema
PESQUISA E INovação



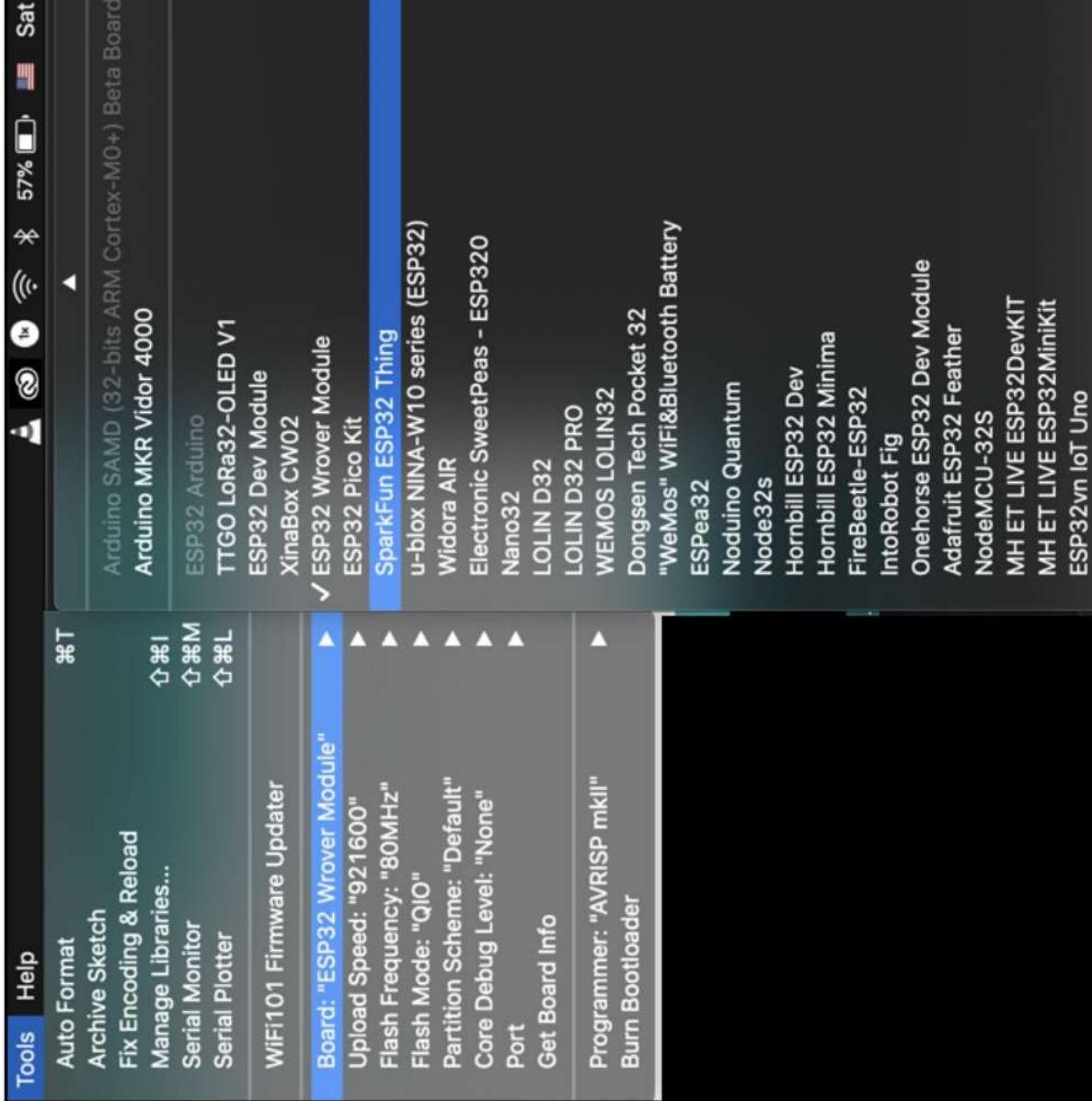
UNIVERSIDADE
ESTADUAL DO CEARÁ



IA

Demo 2 - Fazer um programa Arduino Sketch para ES

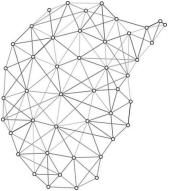
Depois de pronto, você vai
ver a lista de placas ESP32
no software Arduino.



Instituto Iracema
PESQUISA E INovação

UNIVERSIDADE
ESTADUAL DO CEARÁ





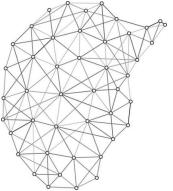
IA

Demo 2 - Fazer um programa Arduino sketch para ES

- Vamos desenvolver o Blinking como um programa Arduino para placas ESP32
- Nossa programa vai rodar apenas um core dos dois presentes no ESP32
 - você pode verificar qual é o core usando a função `xPortGetCoreID()`
 - Vamos ler valores digitais usando a função `digitalRead()`

1. Usando Sketch, vamos relacionar os LEDs aos pinos

```
#define SENSOR_PIR 33 //Define o pino do Sensor PIR
```



IA

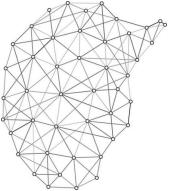
Demo 2 - Fazer um programa Arduino sketch para ES



- Todo programa desenvolvido com ajuda da Arduino IDE contém uma função `setup()` cujo código vai rodar só uma vez no começo.
- Por enquanto vamos configurar para que a leitura do sensor seja escrita na serial

```
void setup() {  
    Serial.begin(115200);  
  
    pinMode(SENSOR_PIR, INPUT);  
  
}  
}
```



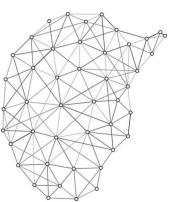


IA

Demo 2 - Fazer um programa Arduino sketch para ES

- O código da função `loop()` roda continuamente como um `while (1)`. Por enquanto, ler o sensor PIR a cada um segundo e imprime na serial se foi detectado algum movimento.

```
void loop() {  
    int status = digitalRead(SENSOR_PIR) ;  
  
    if(status == HIGH) {  
  
        Serial.println("Detectado movimento") ;  
  
    }  
  
    else{  
  
        Serial.println("Nenhum movimento") ;  
  
    }  
  
    delay(1000) ;  
  
}
```



IA

Demo 2 - Fazer um programa Arduino sketch para ES

- Agora, você pode configurar a placa ESP32 alvo e a sua porta é mostrada como na figura ao lado

```
SensorArduinoIDE: Arduino 1.8.19 (Windows Store 1.5.7.0)
Arquivo Editar Sketch Ferramentas Ajuda
Autofornatação Arquivar Sketch
Configurar codificação e recarregar
Gerenciar Bibliotecas...
Sensores/PIR ArduinoID
Sensores/FIM Serial-Serial
Sensores/FIR PrintMode(SPISSOR_E
#define SENSORE_P
void setup() {
  Serial.begin(115200);
  WiFi101 / WiFiNINA_Firmware_Updater
}
void loop() {
  int status = digitalRead(HIGH);
  if(status == HIGH)
    Serial.println();
  else
    Serial.println("Porta: " + String(COM3));
  delay(1000);
}
void (*Setup)(void) = &WiFi101 / WiFiNINA_Firmware_Updater;
void (*Loop)(void) = &loop;
void (*Upload)(void) = &upload;
void (*Program)(void) = &program;
void (*Erase)(void) = &erase;
void (*Reset)(void) = &reset;
void (*BurnFuses)(void) = &burnFuses;
void (*UploadWithReset)(void) = &uploadWithReset;
void (*UploadWithErase)(void) = &uploadWithErase;
void (*UploadWithProgram)(void) = &uploadWithProgram;
void (*UploadWithEraseAndProgram)(void) = &uploadWithEraseAndProgram;
void (*UploadWithResetAndProgram)(void) = &uploadWithResetAndProgram;
void (*UploadWithEraseAndReset)(void) = &uploadWithEraseAndReset;
void (*UploadWithProgramAndReset)(void) = &uploadWithProgramAndReset;
void (*UploadWithEraseAndProgramAndReset)(void) = &uploadWithEraseAndProgramAndReset;
void (*UploadWithResetAndProgramAndReset)(void) = &uploadWithResetAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgram;
void (*UploadWithProgramAndResetAndProgram)(void) = &uploadWithProgramAndResetAndProgram;
void (*UploadWithEraseAndResetAndProgramAndReset)(void) = &uploadWithEraseAndResetAndProgramAndReset;
void (*UploadWithProgramAndResetAndProgramAndReset)(void) = &uploadWithProgramAndResetAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgram;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgram)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgram;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndReset)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndReset;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndReset)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgram;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgram)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgram;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndReset)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndReset;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgramAndReset)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndProgram;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgramAndProgram)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndResetAndProgramAndProgram;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndProgramAndReset)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndResetAndProgramAndProgramAndReset;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndReset)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgram;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgram)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgram;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndReset)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndReset;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndReset)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndReset;
void (*UploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndResetAndProgram)(void) = &uploadWithEraseAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndProgram;
void (*UploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndProgram)(void) = &uploadWithProgramAndResetAndProgramAndResetAndProgramAndProgramAndResetAndProgramAndProgram;
```

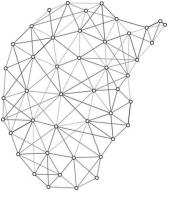
Ctrl+T
Ctrl+Shift+I
Ctrl+Shift+M
Ctrl+Shift+L

ESP3253 Dev Module
ESP32C Dev Module
ESP32 Dev Module
ESP32-WROOM-DA Module
ESP32-Wrover Module
ESP32-HICO-044
ESP32-S3-USG-OTG
ESP3253-CAMLCD
ESP3253-Wrover Kit (all versions)
UM TinyPICO
UM FeatherS2
UM FeatherS2 Neo
UM TinyS2
UM RMP
UM TinyS3
UM PRO3
UM FeatherS3
S.000 Ultra v1
LilyGo-D-Display-S3
microS2
MagicBit

Flash Size: "4MB (32Mb)"
Partition Scheme: "Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "Nenhum"
PSRAM: "Enabled"
Erase All Flash Before Sketch Upload: "Enabled"
Porta: "COM3"
Obter informações da Placa

Desconectar





IA

Material de Apoio

- Kurniawan, A. Internet of Things Projects with ESP32. 2019. Pac Publishing.



UNIVERSIDADE
ESTADUAL DO CEARÁ
Instituto Iracema
PESQUISA E INovação



GOVERNO
BRASIL
MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO
UNIÃO E RE

Dúvidas?

Módulo de Internet das Coisas

Instituto Iracema
PESQUISA E INovação
UNIVERSIDADE
ESTADUAL DO CEARÁ



G
MINISTÉRIO DA
CIÊNCIA, TECNOLOGIA
E INOVAÇÃO

MCTI
FUTURO
Softex

